Average performance of Morris-Pratt-like algorithms

Mireille Regnier

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Average Performance of Morris-Pratt-like Algorithms

Mireille RÉGNIER

N° 2164
Janvier 1994

PROGRAMME 2

Calcul symbolique,
programmation
et génie logiciel

Average Performance of Morris-Pratt-like algorithms

Mireille Régnier

INRIA, 78153 Le Chesnay, France *

Abstract

We propose a general framework to derive average performance of string searching algorithms that preprocess the pattern. It relies mainly on languages and combinatorics on words, joined to some probabilistic tools. The approach is quite powerful: although we concentrate here on Morris-Pratt-like algorithms, it applies to a large class of algorithms, notably to Boyer-Moore-like algorithms. A fairly general character distribution is assumed, namely a Markovian one, suitable for applications such as natural languages or biological databases searching. The average searching time, expressed as the number of text-pattern comparisons, is proven to be asymptotically Kn when the character distribution in the text admits a limit. When the character distributions in the texts and the patterns are both Markovian, the linearity constant is given by a closed formula. In the uniform case, this linearity constant is expressed as a function of the cardinality q of the alphabet.

Analyse en moyenne des algorithmes de Morris-Pratt

Nous proposons un schéma général de calcul de la complexité moyenne des algorithmes de recherche de motifs avec preprocessing du motif. Nous nous appuyons principalement sur les langages et la combinatoire des mots; nous utilisons aussi certains outils de base en probabilités. Cette approche se révèle très puissante: bien que nous nous concentrions ici sur les variantes de Morris-Pratt, elle est d'utilisation générale et s'applique notamment aux algorithmes de la classe Boyer-Moore. Les distributions de caractères sont aussi générales. Nous considérons des distributions markoviennes, adaptées aux applications , comme le langage naturel ou la recherche dans des bases de données biologiques. Nous prouvons que le temps de recherche moyen, exprimé comme le nombre de comparaisons effectuées entre le texte et le motif, est asymptotiquement Kn et calculons la constante de linéarité. Quand les distributions des caractères dans le texte et le motif sont toutes deux markoviennes stationnaires, la constante de linéarité s'exprime par une formule algébrique close. Quand les distributions sont uniformes, ces constantes ne dépendent que de la cardinalité q de l'alphabet.

*This work was partially supported by the ESPRIT III Program No. 7141 ALCOM II.

Average Performance of Morris-Pratt-like algorithms

Mireille Régnier

INRIA, 78153 Le Chesnay, France *

Abstract

We propose a general framework to derive average performance of string searching algorithms that preprocess the pattern. It relies mainly on languages and combinatorics on words, joined to some probabilistic tools. The approach is quite powerful: although we concentrate here on Morris-Pratt-like algorithms, it applies to a large class of algorithms, notably to Boyer-Moore-like algorithms. A fairly general character distribution is assumed, namely a Markovian one, suitable for applications such as natural languages or biological databases searching. The average searching time, expressed as the number of text-pattern comparisons, is proven to be asymptotically Kn when the character distribution in the text admits a limit. When the character distributions in the texts and the patterns are both Markovian, the linearity constant is given by a closed formula. In the uniform case, this linearity constant is expressed as a function of the cardinality q of the alphabet.

Keywords: *string searching, pattern matching, generating functions, analysis of algorithms, automata, complexity, combinatorics on words.*

1 Introduction

This paper is devoted to the evaluation of average performance of string searching algorithms under two probabilistic models: K-order Markov dependencies and Bernoulli stationary distributions. String searching consists

*This work was partially supported by the ESPRIT III Program No. 7141 ALCOM II).

in finding one or all occurrences of some pattern p in a text t, when the pattern and the text both are strings over a same q-alphabet A. The complexity is usually evaluated by the number of character comparisons between the text and the pattern. The worst-case is well known for most algorithms [Riv77, KMP77]. Nevertheless, the worst case is not sure and algorithms are usually expected to behave much better "on the average" [Yao79]. We support this expectation for the main class of algorithms: Morris-Pratt-like [KMP77] and our two probabilistic models. We show in both cases that the expected number of comparisons is asymptotically $K.n$, where n is the size of the text. The linearity constants are also derived. The case of Boyer-Moore-like algorithms, [BM77, BYGR90] is deferred to a companion paper.

So far, quite a few results were available. Attempts in [Sch88, BY89a, BY89b] used Markov chains and hence were limited by the exponential number of states. For a pattern length m greater than $3, 4$, only upper or lower bound could be derived. First asymptotics came out recently. The naive algorithm (respectively Morris-Pratt[KMP77] and Boyer-Moore-Horspool [BM77, Hor80]) were analyzed, for *uniform* character distributions for text and pattern, in [Bar85] (respectively [Rég89] and [BYGR90, BYR92]). Nevertheless, assumptions of a uniform distribution of characters hardly holds for numerous applications, notably the search in a text written with some natural language or biological applications. First results for biased stationary distributions are provided in [Sch88, BY89a, BY89b], that were based on probability, namely Markov chains . Nevertheless, for pattern lengths m greater than $3, 4$, only upper or lower bound could be derived. This is due to a combinatorial explosion of the number of states when the size m of the pattern increases. The algebraic approach of [Rég89], using combinatorics on words, is extended to biased stationary distributions in [Rég91]. It allows very precise asymptotic developments of the linearity constant. Some results on Morris-Pratt can also be found in [Han91]. The combinatorial explosion stucks them to small patterns or asymptotic order and ,a fortiori, prevents from a generalization to Markovian dependencies. A first generalization to such distributions is achieved in [Rég92b], for Morris-Pratt like algorithms and Boyer-Moore-Horspool. It is based on a language approach. This paper groups all these results.

Our approach is algebraic. As a matter of fact, the main problem for the analysis is the huge number of states of the automaton defining each algorithm: q^m where m is the size of the searched pattern and q the cardinality of the alphabet. We propose a general framework based on languages and combinatorics of words, that drastically reduces the combinatorial explo-

sion of the problem. We propose a *bootstrapping* approach. We concentrate first on the main contributions, defined by a partition of the states into a few number of states, and refine later. Practically, the convergence to the actual cost is very fast, and we provide a *closed formula* for the Morris-Pratt algorithms. Moreover, one can consider simultaneously all possible lengths m. More precisely, performance show a simple dependancy on this parameter and are simply obtained, for finite m, from the general result by truncation of a Taylor development. We reduce the performance analysis to word enumeration. We make use of algebra, mainly generating functions and combinatorics on words, notably a canonical representation of periods. Nevertheless, probability theory is still useful, notably for Boyer-Moore analysis. Our scheme applies for a large class of algorithms, namely the ones based on a preprocessing of the searched pattern, and fairly general distributions. More precisely, we assume 1-order Markovian dependencies between the characters. This is suitable for applications to natural languages or molecular biology. One interesting property of our approach is that the computation cost for all variants steadily derives from the cost computation of any one of them. Here, we chose Morris-Pratt variant as the initial algorithm. Then, we provide a closed formula, valid for any m, for the difference with the linearity constant of Knuth-Morris-Pratt variant. The scheme easily extends to Simon variant. We prove that the average searching time, expressed as the number of text-pattern comparisons, is linear in the size n of the text and compute the linearity constant. Some open problems are also pointed out.

2 Morris-Pratt-like Algorithms

We can now define formally the "Morris-Pratt-like" algorithms [KMP77]. Remark at first that string searching algorithms are universally based on a couple of variables (AP, PP) , namely the alignment position and the pattern position, and they perform text-pattern comparisons: $t[AP]?p[PP]$. The only differences -that may be important!- rely in the way these variables are updated.

Definition 1 *A Morris-Pratt-like algorithm is a string searching algorithm where the basic couple maintains property (P1):*

Morris-Pratt property (P1)

(1) AP is increasing, with a difference upper bounded by the pattern size.

(2) Characters are read sequentially.

As a corollary, the only comparisons authorized satisfy:

$$PP = \max\{k; t[AP] \dots t[AP + k - 2] \preceq p\}$$

This definition is very closed to the definition of "Sequential algorithms" given recently in [Han93]. Interestingly enough, [CGG90] algorithms satisfy the last property in their first step.

Let us make this precise. The Morris-Pratt-like algorithms read the text from left to right, by condition (2). This is an advantage over the naive algorithm, as they never read backward. Condition (2) ismaintained in the following way. After a match, both pointers move one step forward, except when the pattern is found. After a mismatch, or when p is found, the new alignement is determined by the largest side p'' of the prefix $p' \preceq p$ already found, i.e. the largest subsequence p'' such that p'' is a strict prefix and suffix of p'. This new alignment is determined from a pattern preprocessing. A *next* function is defined, for any j: p' being the prefix of length j of p, p'' its largest side, of length $k - 1$, then $next[j] = k$. This means that the first character of p is now aligned on the first character of the right border of p'. Remark that whenever the value of PP is 1, PP remains equal to 1 after a mismatch. For example, let

$$p = 01201201345$$

$$t = 03012012011012012013 45$$

A first match occurs, PP and AP are shifted simultaneously to the right. The mismatch between $t[2] = 3$ and $p[2] = 1$ implies to move PP back to $p[1]$ and the next mismatch: $t[2] \neq p[1]$ implies that PP and AP shift simultaneously: $PP = p[1], AP = t[3]$. Then, five matches occur, followed by a mismatch, $t[1] = p[9]$. The largest side of $p' = 01201201$ is 01201, hence PP becomes 6. As $p[7] \neq t[11]$ and 01 is the largest side of 01201, PP becomes 3. Finally, $p[3] \neq t[11]$ leads to $PP = 1$ and a mismatch. AP shifts by 1, and the pattern is found.

It is worth noticing that this algorithm does not take into account any information on the mismatching character. As a matter of fact, when $next[j]$ is k, the next comparison to be performed is $t[AP]?p[k]$, whose result is known to be false whenever $p[j] = p[k]$. This occurs in our example for

$j = 6$: $next[6] = 3$ and $p[3] = p[6]$. Hence, $next[j]$ may be redefined as the largest k such that $p[1]\cdots p[k-1] = p[j+1-k]\cdots p[j-1]$ **and** $p[j] \neq p[k]$, that is precisely the Knuth-Morris-Pratt option.

3 Probabilistic Tools

3.1 Probabilistic models

We consider here three probabilistic models: stationary distributions, Bernoulli stationary distributions and k-Markov dependencies and define them formally. At first, let us remark that pattern and text both vary. Hence, one must define two probability measures on A^*: one, Π is associated to the text, the other, $\mathcal{P}$, is associated to the pattern.

Stationary distributions

Definition 2 *A text string $(X_i)_{i\in N}$ is said stationary iff it is a stationary sequence.*

Stationary Bernoulli model Here, we assume independency between the different positions in the text or the pattern.

Definition 3 *We note $\{p_a\}_{a\in A}$ (respectively $\{q_a\}_{a\in A}$) the character distributions in the pattern (respectively the text). That is, for any position in the pattern or the text, the probability that it is occupied by a given character a is: p_a (respectively q_a). When all p_a (or q_a) are equal, the model is said uniform, otherwise it is biased. We note*

$$\begin{cases} s_{j,i} = \sum_{a\in A} p_a^j q_a^i, & 1 \leq i \neq j \\ \sigma_i = \sum_{a\in A} (p_a q_a)^i = s_{i,i}\ , & 1 \leq i \end{cases}$$

Remark: We assume notably that the probability of occurrence of a character a at some position does not depend on the neighbours. Our parameter σ_1 coincides, when $p_a = q_a$, with the parameter p_{equal} considered in [BY89a] that reduces to $\frac{1}{q}$ for uniform text and pattern distributions.

k-order Markov dependencies The model above is too rough for many applications. In natural languages or in DNA sequences, the frequency of a character depends on the neighbours. For instance, in French language, a

u is almost certain after a q. Let us present k-order Markov dependencies for infinite sequences of random variables $(X_l)_{l\in N}$. One assumes that the distribution of values for random variable X_l only depends on the values taken by the k random variables $X_{l-k},\ldots,X_{l-1}$. For $k=1$, this leads to define, for any couple $(a,b)\in A^2$:

$$p_{a,b} = Pr(X_l = b/X_{l-1} = a) \ .$$

In our presentation, we will assume, for sake of clarity, that $k=1$. The generalization to $k>1$ is deferred to the Appendix. That is:

Definition 4 *We assume that the text and the pattern define two Markov processes on A^*, with transition matrices $Q = ||q_{i,j}||$ and $P = ||p_{i,j}||$. Let $\mathcal{Q}$ and $\mathcal{P}$ be the probability measures so defined on A^*. We assume both processes admit a limiting process, and note $(q_i)_{i=1\ldots q}$ and $(p_i)_{i=1\ldots q}$ the stationary probabilities.*

Then, w being a subword in the text (or the pattern), we have:

$$E_{\mathcal{Q}}(w[k+1] = a_j/w[k] = a_i) = q_{i,j} \ .$$

Moreover, if w is "far enough" to the right in the text: $E_{\mathcal{Q}}(w[1] = a_i) = q_i$. If the size $|m|$ of the pattern is "big enough" and w a subword starting "far enough" to the right: $E_{\mathcal{P}}(w[1] = a_i) = p_i$.

3.2 Parameters of evaluation

It is generally admitted to estimate the complexity of string searching algorithms as the number of text-pattern comparisons. Definition 5 below formalizes the intuitive computational approach followed in previous works. As two parameters range simultaneously -the text and the pattern-, one must be careful to the way they simultaneously tend to infinity.

Definition 5 *Given two words p and t and an algorithm A, we note $C^A(t,p)$ the number of comparisons performed when pattern p is searched in text t using algorithm A. Let $C_n(p)$ be the average value, over the set of texts of size n, with a given character distribution Π, of $C(t,p)/n$. We also define, when it exists:*

$$C(p) = \lim_{n\to\infty} C_n(p) \ .$$

$C(p)$ is called the p-linearity constant. *Then, the average value of $C(p)$, when p ranges over all patterns of length m, is well defined:*

$$C_m = E_{\mathcal{P}}[C(p)/|p| = m] \ .$$

Finally, let

$$C = \lim_{m \to \infty} C_m$$

when this limit exists. The algorithm is said asymptotically linear and C is called the linearity constant *of the algorithm.*

The existence of such limits are not ensured and depends on the algorithm and the text and pattern distributions. Also, the definitions of $C(p)$ (respectively C) are meaningful only for text (respectively pattern) distributions that admit a limit, e.g. stationary processes: these are the minimal assumptions. With a finite alphabet, $C_n(p)$ is a finite sum, hence well defined. The existence of the limits $C(p)$ and C are not guaranteed. Nevertheless, when $C(p)$ exists, C_m is a finite sum, hence exists. Our derivation of the expectations above will imply, for these specific algorithms:

$$\forall p \quad C_n(p) \to_{n \to \infty} C(p)$$

And we will formally prove the convergence of C_m to a constant C.

The justification of this computation order is that the algorithms considered are based on a *preprocessing* of the pattern, and not of the text. Then the m characters of the pattern are compared in turn to some characters in the text and, when a mismatch or an equality occurs, algorithmic decisions depend on the pattern. Remark that the dual approach is also possible: compute first the performances for a given text and a random pattern, and then average over the texts. Final results of these two specific orders are equivalent. An interesting case occurs when $\frac{C^A(t,p)}{|t|}$ converges a.s. and in expectation to some random variable C^A. Then, the *existence* of the linearity constant can be proven, that is still *computed* by the method above. This will be considered in a companion paper [RS94].

One expects the linearity constants to depend on the cardinality q of the alphabet, and, if the distribution is not uniform, on the data distributions. For an intricate function, one may give approximations. In works dealing with uniform distributions [BY89a, Rég89, BYGR90], one provides a Taylor development in $\frac{1}{q}$. We formalize below the approximation we used in our preliminary work [Rég91]:

Definition 6 *A k-approximation of the linearity constant of an algorithm is a formula:*

$$C_m = \phi_m(\Pi, \mathcal{P}) + O(\epsilon^k)$$

where ϕ_m and ϵ are two functions of the text and pattern distributions, assumed to be Bernoulli stationary, and ϵ is independent of m.

Let us remark it is necessary that the approximation term be smaller than the equivalent function ϕ, To guarantee it (when possible), we need to have precise knowledge on the constant. We will return to that point in the last section.

4 Methods

4.1 State of the Art

We describe various attempts. The first difficulty is the definition of the *meaningful parameter*. In the first work on average performance, [Bar85], where Knuth-Morris-Pratt is considered, one searches the *first occurrence* of the pattern. A full match is then an **absorbing state** and the parameter of interest, the number of comparisons to be performed, is claimed to be equal to the **expectation of the absorbing state**, defined as the number of steps the process makes from a start until absorption. This led to a paradoxal result, where the naive algorithm appeared asymptotically better than Knuth-Morris-Pratt, while Knuth-Morris-Pratt, by construction, always outperforms the naive algorithm for any pattern and text...The reason is that the claim is true for the naive algorithm, false for Knuth-Morris-Pratt. Details of the refutation may be found in [Rég89].

All other analyses consider the average number of comparisons to find *all occurrences* of a given pattern. Remark that we consider algorithms that are based on a *preprocessing of the pattern* and not of the text, as for instance the one in [KR87]. Such algorithms are currently seen as a set of automata $\{A_p\}$ [HU79, Tho68]: automaton A_p recognizes in a text t, the input, all the occurrences of pattern p. Each text-pattern comparison leads to a state change, that also determines the next comparison to be performed. Hence, the complexity is measured by the number of transitions. One approach has been to use Markov chain. One writes down for every p the associated automaton [Bar85, BY89a, Han91] and compute its steady state, hence $C_p(Alg)$. Let us consider the problems that occur. A crucial parameter, for Knuth-Morris-Pratt, is the number s of characters read since

the last mismatch. One out of these is compared twice to the pattern which means that $s+1$ comparisons are necessary to "get rid" of s characters. Hence, the contribution to the cost is: $(s+1)/s$ which leads to the average cost:

$$1 + E\left(\frac{1}{s}\right) .$$

This parameter is also called the **shift**. It is also crucial for many string searching algorithms, as discussed below. The first problem cis that $1/s$ is not a natural parameter. Second, Markov chains, when exact, are **not homogenous** in space. I.e. different patterns define different automata and Markov chains. It is possible to study the *steady states* for each subchain but, when the length m of the pattern increases, the number of possible chains increase rapidly. The model quickly becomes untractable. As a matter of fact, an attempt by this method is derived in [Han89] that is stopped by the complexity of computation at $m = 5$. In a more recent work [Han91], one gets to $m = 10$ for uniform distributions, but only to $m = 4$ for biased ones. In both cases, the linearity constant is known to order 4 at most. This combinatorial limitation would be even more severe for more intricate distributions, such as k-Markov dependencies. Approximations have also been tried. [BY89a] describes an **embedding in a homogenous** Markov chain weighted by probabilities. Such an approximation is mathematically illegal: the steady state is usually an inexact approximation of the steady states of non homogenous Markov chains. Notably, the author in [BY89a] is unable to prove his approximation by a linear number of states. As a matter of fact, as seen in Section 8, our results coincide only up to order 1.

4.2 Languages and automata

We now briefly introduce our approach. As extensively discussed in [Rég89, Rég91], the combinatorial explosion of the number of states quickly makes the approach via Markov chains untractable. We advocate in this paper that string searching performance evaluation reduces to word enumeration and should rely on combinatorics on words. To avoid the combinatorial explosion, our aim is a simultaneous computation of the contribution of several states of the automaton. We will *classify* the possible states by a bootstrapping approach in such a way that the first classes provide the main contribution. In the specific case of Morris-Pratt-like algorithms, the computational process converges nicely and yields a closed formula. To make the computation easier, we use the powerful tools on languages: we follow

[Eil74] and associate a word w to each state of the automaton $\mathcal{A}_p$. Hence a language $\mathcal{L}_p^{(i)}$ is associated to each class. Also, we define a cost function ϕ for each occurrence of $w \in \cup\mathcal{L}_p^{(i)}$ **in the text**. I.e. the sum

$$\sum_{w \in \mathcal{L}_p} E_{\mathcal{Q}}(w)\phi(w) \tag{1}$$

where $E_{\mathcal{Q}}(w)$ is the probability for w to occur in the text yields $C(p)$, while the sum

$$\sum_{p \in A^m} E_{\mathcal{P}}(p) \sum_{w \in \mathcal{L}_p} E_{\mathcal{Q}}(w)\phi(w) \tag{2}$$

yields the linearity constant C. Our main constraint on languages $\mathcal{L}'_p$ is the easiness of computation of (1) and (2). They will be defined from basic languages via word constructors such as concatenation, exponentiation,...Also, our language choices will rely on the following fundamental remark. Any backward edge to any but the initial state is associated to a self-overlapping prefix p' of p. I.e. a prefix that factorizes $p' = ua = bu$. u is called a *border*. This equation has been extensively studied in [Lot83], and defines periodic words p'. We recall in the appendix some basic results in combinatorics on words. Also, we define the generating functions as well as the now classical computational tools [VF90].

5 Bootstrapping computation of the Morris-Pratt language

Our approach is based on the *memoryless property of Knuth-Morris-Pratt.* I.e. given a position in the text, the number of comparisons performed at that position only depends on a few left neighbours, at most $|p|$. Hence, we will characterize the subsets of preceding words inducing extra-comparisons, in order to get a fast convergence of the expression in (1). We first introduce the basic notion of quasi-mismatch.

Definition 7 *Given p, a* quasi-mismatch *is a word $p'b$ such that:*

$$p' \in A^{*+}, b \in A \text{ and } p' \preceq p, p'b \not\preceq p \ .$$

A false-mismatch *is a quasi-mismatch that is a proper suffix of a p-prefix. Such a prefix is called a* cover. *A cover is said* minimal *if it is the smallest cover of a quasi-mismatch. Let $\mathcal{L}_p^{(1)}, \mathcal{L}_p^{(2)}$ and $\mathcal{N}_p^{(3)}$ be the languages associated to quasi-mismatches, false-mismatches and covers.*

These notions are of particular interest, as each Morris-Pratt-like algorithm defines a unique surjection from the set of extra-comparisons into the set of quasi-mismatches that are not false-mismatches. More precisely, any extra-comparison implies a mismatch, that implies itself a quasi-mismatch. I.e. for all variants, an alignment of the text with a proper suffix p' of p, followed by one character b such that $p'b \not\preceq p$ occurs. Also, a quasi-mismatch is not a mismatch iff it belongs to a (greater) matching sequence. That is a false quasi-mismatch according to Definition 7. For example, with $p = 012012013$, let $t = **012012013**$, two quasi-mismatches 013 and 012013 are counted for 3, which belongs to a matching sequence. Both are suffixes of a p-prefix. Minimality property plays a role in enumeration, which leads to define our language $\mathcal{N}^{(3)}$. We now may characterize the language of covers and quasi-mismatches.

Theorem 5.1 *Given a pattern p, the subset of quasi-mismatches is the set $\mathcal{L}_p^{(1)}$:*

$$\mathcal{L}_p^{(1)} = (A + \cdots + A^{m-1}) \cap \{p\text{-}prefixes\} \cdot A - (A^2 + \cdots + A^m) \cap \{p\text{-}prefixes\} \ .$$

The set of false-mismatches is $\mathcal{L}_p^{(2)}$ defined by:

$$\begin{cases} L^{(2)}(b,\bar{b}) & = \{u(vu)^l b; \bar{b} \preceq vu, u \neq \epsilon\} \\ L^{(2)} & = \cup_{b \in A} L^{(2)}(b) \\ \mathcal{L}_p^{(2)} & = L^{(2)} \cap \{p\text{-}prefixes\} \ . \end{cases}$$

Proof: The expression of $\mathcal{L}_p^{(1)}$ is a straightforward consequence of the definition. Now, a quasi-mismatch is associated to a prefix $p''b$ such that p'' is selfoverlapping: p' is simultaneously prefix and suffix. Applying classical results on selfoverlapping or periodic words [Lot83], we get $p'' = uvu$ that rewrites $u(vu)^l, l \geq 1$. This yields $\mathcal{L}_p^{(2)}$.

Theorem 5.2 *Let us define*

$$\begin{cases} L^{(3)}(b,\bar{b}) & = \{u(vu)^l v_2 u(vu)^l b; b \preceq v_2 u(vu), \bar{b} \preceq vu, u \neq \epsilon\} \ , \\ L^{(4)}(b,\bar{b}) & = \{a(wa)^l[(wa)^m va(wa)^{l+m}]^2 b; a \neq \epsilon, wa \in P, \bar{b} \preceq va, b \preceq wa\} \ . \end{cases}$$

Also, we denote for $i \in \{3,4\}$, and a given pattern p:

$$\begin{cases} L^{(i)} & = \cup_{b,\bar{b} \in A} L^{(i)}(b) \\ L_p^{(i)} & = L^{(i)} \cap \{p\text{-}prefixes\} \ . \end{cases}$$

Given a pattern p, a cover is non-minimal iff it belongs to $\mathcal{L}_p^{(3)} \cup \mathcal{L}_p^{(4)}$.

Proof: Given a quasi-mismatch u embedded into two greater false-mismatches f_1 and f_2, we have [Lot83] $f_1 = u(vu)^l$. Assume that $f_2 \neq f_1 v f_1$. Then $u(vu)^l$ self-overlaps and the k-overlapping lemma in [Rég92b] implies that $f_2 = f_1.p_i^*$, where p_i is either vu or some canonical border of u. Reciprocally, assume u is a non-bordered word $a(wa)^{l+m}, m \geq 1$ and denote $vu = va(wa)^{l+m}$. Then $a(wa)^l[(wa)^m va(wa)^{l+m}]^2.b$ qualifies to be a greater false mismatch. □

All this formalizes in language terms the reasoning of [Rég89, Rég91]. We are now ready to state (1) for Morris-Pratt algorithm.

Theorem 5.3 *With the notation above:*

$$C^{MP}(p) = 1 + \sum_{w \in \mathcal{L}_p^{(1)}} E_Q(w) - \sum_{w \in \mathcal{L}_p^{(2)}} E_Q(w) + \sum_{w \in \mathcal{L}_p^{(3)}} E_Q(w) + \sum_{w \in \mathcal{L}_p^{(4)}} E_Q(w) \ .$$

Example: Let us show on one example how this allows quick evaluation of the cost for a given p. Let, say, p be 01020. We denote S the set of its prefixes, i.e. we have $S = \{0, 01, 010, 0102\}$. Now, we get:

$$\mathcal{L}_p^{(1)} = \{0, 01, 010, 0102\}.A - \{01, 010, 0102, 01020\} \ .$$

Now, $\mathcal{L}_p^{(2)}$ is the subset $\{0102\}$ of S where $u_1 = 0, v_1 u_1 = 10, a = 2$. Finally, among the set of strict prefixes $\{0, 01, 010, 0102\}$ we obtain, no word has several periods, i.e. no word belongs to $\mathcal{L}_p^{(3)}$, that is: $\mathcal{L}_p^{(3)} = \emptyset$. Applying formula 5.3, , we get: $C'^{MP}(p) \equiv 1 + \sum_{w \in \mathcal{L}_p^{(1)}} E_Q(w) - E_Q(0102)$. For uniform distributions, this leads to: $1 + (\frac{1}{q} - \frac{1}{q^5}) - \frac{1}{q^4}$. Remark that the simplification of $\sum_{w \in \mathcal{L}_p^{(1)}} E_Q(w)$ is general. This will be detailed below in 6.1.

Proof: To state this expression, analogous to (1), we need define ϕ functions. Note first that every character is read at least once by condition (3), which yields the first term. Additionnally, for Morris-Pratt variant, all mismatches imply extra-comparisons, except for the ones occurring on the first character. I.e. we may chose $\phi(w) = 1$ if we only count actual mismatches. Such mismatches can be derived from our basic languages $L_p^{(i)}$. As a mismatch implies a quasi-mismatch, $\mathcal{L}_p^{(1)}$ provides an upper bound. We now enumerate false mismatches. We consider the primitive word x associated to vu, and rewrite $p' = zx^*, x^* = zx^l$, with $z \preceq x$. The condition $b \not\preceq vu$ still holds (i.e. we have $b \not\preceq x$), and it prevents $z.x^{l+1}$ to be a prefix and a false mismatch.

Hence ϕ should be l when we consider this *unique* decomposition based on primitive words. Nevertheless, we can derive from it an other unique decomposition, easier to compute. Consider $k \in [1 \ldots l]$. By division, we get: $l = nk + r, r < k$. Then, we may rewrite:

$$u(vu)^l = ((uv)^r u)((vu)^k)^n = ((uv)^r u).(w.((uv)^r u))^n = u'.(v'u')^n \ .$$

As we have l such decompositions, this allows to take rid of the condition $vu \in P$ while changing $\phi(w) = l$ into $\phi(w) = 1$. Finally, in order to substract false-mismatches only once, we must consider only minimal covers, or, equivalently, add non-minimal covers. □

Remark also that the definition of our languages totally relies on the periodicities of p-prefixes. One knows how to derive all prefixes in linear time. As all periods of each prefix can be derived simultaneously, one can trivially compute the cost for Morris-Pratt algorithms in quadratic time, for any given pattern, and any distribution. Remark this is closely related to superprimitivity testing [Bre93]. Nevertheless, one is usually more interested by the average cost, when p ranges. This derivation of the linearity constant is presented in the next section.

6 Morris-Pratt linearity constant

We detail the performance of Morris-Pratt algorithm, from which performance of Knuth-Morris-Pratt variant follow. Linearity constant derives from a computation of $L^{(i)}$ languages expectations. I.e. Theorem 5.3 translates into:

Theorem 6.1 *Averaging over all patterns p of size m, we get:*

$$C^{MP} = 1 + L^{(1)}(\{p_q a q_a\}) - L^{(2)}(\{p_q a q_a\}) + L^{(3)}(\{p_q a q_a\}) + L^{(4)}(\{p_q a q_a\})$$

from which C^{MP}_m derives by a truncation at order m.

6.1 Computing language expectations

Lemma 6.1 *Let $\mathcal{L}$ be the language $\cup\{a(va)^k; a \in A^* - \{\epsilon\}, v \in A^*, c \preceq va\}$ where c is a given character in A and k an integer. The generating function of $\mathcal{L}$ is:*

$$z_c^k F(\{z_a^k\})F(\{z_a^{k+1}\}) + z_c^{k+1} F(\{z_a^{k+1}\}) - z_c^k F(\{z_a^k\})$$

Proof: One considers in turn $v = \epsilon$ and $v \neq \epsilon$ and the result steadily follows from our basic rules. Remark that $\mathcal{L}$ contains all words in A^* with all their prefixes (respectively all their prefixes and extensions) when k is 1 (respectively k ranges over N).

□

Main contribution: $L^{(1)}$ Our first result is a general formula for $\mathcal{L}_p^{(1)}$. It was derived in [Rég89, Rég91] and generalized to Markovian dependencies in [Rég92a]:

Proposition 6.1 *For Bernoulli stationary distributions:*

$$\sum_{w \in \mathcal{L}_p^{(1)}} E_Q(w) = E_Q(p[1]) - E_Q(p)$$

Proof: One has $\sum_{b \in A} E_Q(b) = 1$. Also, $A^j \cap \{p - \text{prefixes}\}$ reduces to the unique prefix of p of size j. This yields the first equation by elimination of the median terms. Notice that for uniform distributions, this leads to: $\frac{1}{q} - \frac{1}{q^m}$.

First correcting term: $L^{(2)}$ From now, we only consider the stationary case and use generating functions. Equivalent results for Markovian dependencies will be considered in 9.

Proposition 6.2 *Given a stationary distribution, one has:*

$$L^{(2)}(\{z_i\}) = \frac{t_2({t_1}^2 - t_2)}{(1 - t_1)(1 - t_2)} + \sum_{l \geq 1} \frac{t_1 t_l - -t_{l+1}}{(1 - t_l)(1 - t_{l+1})}$$

For uniform stationary distributions, this simplifies into

$$L^{(2)}(z) = (q - 1)\frac{z^2}{1 - z}(W(z) - 1) .$$

Proof: Applying 6.1 yields the generating functions:

$$\sum_{l \geq 1} [T(t_{l+1}) - 1].T(t_l).(t_1 t_l - t_{l+1}) + \sum_{l \geq 1} T(t_{l+1}).(t_1 t_{l+1} - t_{l+2})$$

that rewrite $\sum_{l\geq 1} T(t_{l+1}).T(t_l).(t_1t_l - t_{l+1}) - T(t_1)({t_1}^2 - t_2)$. Remark that the convergence of the infinite sum is ensured, as $t_l = O(q^{-(2l-1)})$. This expression greatly simplifies in the uniform case, as:

$$\begin{cases} t_1t_l - t_{l+1} & = (q^2-q)z^{l+1} \\ z^{l+1}.\frac{1}{1-t_{l+1}}.\frac{1}{1-t_l} & = \frac{z}{z-1}(\frac{z^{l+1}}{1-qz^{l+1}} - \frac{z^l}{1-qz^l}) \end{cases}$$

and we get $(q-1).\frac{z^2}{1-z}.[W(z)-1]$, which is G. Finally, truncation at order m yields the desired contribution. □

Last refinement: $L^{(3)}$ and $L^{(4)}$

Proposition 6.3 *Assume a Bernoulli stationary distribution for the text and the pattern. Denote $L^{(3)}(\{z_i\})$ the multivariate generating function associated to $L^{(3)}$. We have:*

$$\begin{aligned} L^{(3)}(\{z_i\}) \;=\; & \sum_{l\geq 1}(t_{2(l+1)}t_2 - t_{2(l+2)})T(t_1) + \sum_{l\geq 1}(t_{2(l+1)}t_{2l} - t_{4l+2})T(t_{2l}) \\ & + \sum_{l\geq 1}(T(t_{2(l+1)})-1)(t_{2(l+1)}t_2 - t_{2(l+1)})T(t_1)T(t_{2l}) + (t_{2(l+1)}t_2 - t_{2(l+2)})T(t_1) + (\end{aligned}$$

In the uniform case, this simplifies into:

$$q(q-1)[\sum_{l\geq 1} z^{2l+4}W(z) + z^{4l+3}W(z^{2l}) + \sum_{l\geq 1}(W(z^{2l+2})-1)[z^{2l+4}W(z) + z^{4l+7}W(z^2l) + z^{2l+2}W(z)W(z$$

Proof:

Let b be some given character in A. Denote $\bar{b}$ any character in $A - \{b\}$. Let l be fixed. Applying our translation rules yields, when $v, v_2 \neq \epsilon$ the general term $(W(z^{2l+2}) - 1)z_{\bar{b}}^{2l}W(z^{2l}).z_bW(z).z_b$. Now $v = \epsilon \Rightarrow b \not\preceq u \Rightarrow v_2 \neq \epsilon$ while $v_2 = \epsilon \Rightarrow b \preceq u \Rightarrow v \neq \epsilon$. These disjoint cases provide the two terms $z_{\bar{b}}^{2l+2}W(z^{2l+2})z_b^2W(z)$ and $z_b^{2l+2}W(z^{2l+2})z_{\bar{b}}^{2l}W(z^{2l})z_b$ which appear to be the main contribution. Summing yields the two expressions above. Notice, that for any integers k and l, we have $E_{\mathcal{QP}}(\bar{b}^l.b^k) = t_lt_k - t_{k+l}$. □

Proposition 6.4 *Assume a Bernoulli stationary distribution for the text and the pattern. Denote $L^{(4)}(\{z_i\})$ the multivariate generating function associated to $L^{(4)}$. We have:*

$$L^{(4)}(\{z_i\}) = -\frac{t_5t_2 - t_7}{(1-t_4)(1-t_2)} + \frac{1}{1-t_2}\sum_{k\geq 0}\frac{t_2t_{5+3k} - t_{7+3k}}{(1-t_{4+3k})(1-t_{7+3k})}\ .$$

In the uniform case, this simplifies into:

$$L^{(4)}(z) = (q-1)\frac{z^7}{1-z^3}(W(z^4)-1).W(z^2) \ .$$

Proof: We now extend the proof of 6.2. We rewrite $a(wa)^l = a(wa)^r(wa)^{km}$, with $r < m$. Hence, $(wa)^m$, which may be any non-empty word in A^* is associated to any of its prefixes or extensions. In the uniform case, all extensions contribute by $\sum x^j = \frac{x}{1-x}$. As $(wa)^m$ (respectively $a(wa)^l$) occurs 4 times (respectively 3 times), we get $[W(z^4)-1].\frac{x^3}{1-x^3}$. Then b is determined and $q-1$ choices remain for $\bar{b}$.

In the biased case, we consider the set of configurations $\cup\{u^{4+3k+3}v^{4+3k}.b\}_{k\geq 0, u\neq\epsilon, b\preceq vu}$. We use again 6.1. The sum of the last two terms simplifies into $-z_b^5 F(\{z_a^4\})$, which yields, combined with $z_{\bar{b}}^2 F(\{z_a^4\})$ our first term. We derive the main term from the first term. □

7 Knuth-Morris-Pratt variant performance

We only need to correct slightly our languages $\mathcal{L}^{(i)}$. To change Morris-Pratt into Knuth-Morris-Pratt, one deletes from $\mathcal{L}^{(i)}$ the words w such that the associated quasi-mismatch $w\bar{b}$ is a ***skipped*** mismatch. I.e. the largest border p' of w is followed by $\bar{b}$. Hence, if character b is found in the text, comparison p'? will not be performed. We consider in turn the two disjoint cases:

$$\begin{aligned} p' = \epsilon &\Leftrightarrow w \in S, \bar{b} \preceq w \\ p' \neq \epsilon &\Leftrightarrow w = u(vu)^k, u \neq \epsilon, \bar{b} \preceq vu \end{aligned}$$

The reasoning in Section 5 applies to provide the expressions of the correcting sets $\mathcal{M}^{(i)}$. They will depend on the set of non-bordered words.

Lemma 7.1 *We define, for $i \in \{1,2,3\}$, the set of languages:*

$$\begin{aligned} \mathcal{M}^{(1)}(\bar{b}) &= \{u; \bar{b} \preceq u, u \in S\} + \cup_{k\geq 1}\{u(vu)^k; u \neq \epsilon, \bar{b} \preceq vu, vu \in P\} \\ \mathcal{N}^{(1)}(\bar{b}) &= \cup_k\{u(vu)^k v_2 u(vu)^k \bar{b}; u \neq \epsilon, \bar{b} \preceq vu, v_2u(vu) \neq (vu)^l\} \\ \mathcal{M}^{(2)}(b,\bar{b}) &= \{w_1w_2w_1b; w_1 \in M_{\bar{b}}^{(1)}, \bar{b} \preceq w_2w_1\} + \cup_{k\geq 2}\{u(vu)^k b; u \neq \epsilon, \bar{b} \preceq vu \in P\} \\ \mathcal{N}^{(2)}(b,\bar{b}) &= \cup_{k\geq 1, j\geq 2}\{u[(vu)^k v_2 u(vu)^k]^j b; \bar{b} \preceq vu, v_2u(vu) \neq (vu)^l\} \\ \mathcal{M}^{(3)}(b,\bar{b}) &= \cup_{w_2b\in\mathcal{M}^{(2)}(b,\bar{b})-\mathcal{N}^{(2)}(b,\bar{b})}\{w_2.w_3w_2.b; b \preceq w_3w_2\} \end{aligned}$$

We denote $\mathcal{M}^{(i)} = \cup_{b,\bar{b}}\mathcal{M}^{(i)}(b,\bar{b})$ and $\mathcal{M}^{(1)} = \cup_{\bar{b}}\mathcal{M}^{(1)}(\bar{b})$. Also, we set $\phi(w) = \lfloor\frac{k}{2}\rfloor$ for the second subset of $\mathcal{M}^{(2)}$ and $\phi(w) = 1$ otherwise.

We can now express the language associated to Knuth-Morris-Pratt variant:

Theorem 7.1 *The linearity constant of the Knuth-Morris-Pratt algorithm, C_m^{KMP} satisfies;*

$$-C_m^{MP} + C_m^{KMP} = -\sum_{\bar{b}} \sum_{w \in \mathcal{M}_{\bar{b}}^{(1)}} E_{\mathcal{Q}}(wb) E_{\mathcal{P}}(w\bar{b}) + \sum_{\bar{b}} \sum_{w \in \mathcal{N}_{\bar{b}}^{(1)}} E_{\mathcal{Q}}(wb) E_{\mathcal{P}}(w\bar{b}) + \sum_{w \in \mathcal{M}^{(2)}} E_{\mathcal{QP}}(w)\phi(w) - \sum_{w \in \mathcal{N}^{(2)}} E_{\mathcal{QP}}(w) - \sum_{w \in \mathcal{M}^{(3)}} E_{\mathcal{QP}}(w) .$$

Proof: From the two disjoint cases defined above, $M^{(1)}$ contains all skipped mismatches. Whenever u is a non-bordered word or vu is a non-degenerated period (i.e. $k \geq 2$), $p' = \epsilon$ or $p' = u(vu)^{k-1}$ is the largest border. A contrario, if q is multiperiodic, e.g. $q = u(vu)^k = u'(v'u')$ with $|u'| < |u|$, the smallest one is not to be counted. By ??, $u = \alpha(\beta\alpha)^m = u'(\beta\alpha)^*$. Using 5 yields $N^{[1]}$, and we have the correcting set for $L^{(1)}$.

We now correct $L^{(2)}$. Let p_2 be a false mismatch associated to an (unskipped) quasi-mismatch p_1. p_1 is a border of p_2. If it does not self-overlap in p_2, we rewrite $p_2 = p_1 w p_1$. Remark this occurs notably when p_1 is in S. Otherwise, we have: $p_2 = u(vu)^k$ with $p_1 = u(vu)^l, 1 \leq l < k$. This leads to $\mathcal{M}^{(2)}$. Also, if $u(vu)^k b \preceq p$, all $u(vu)^l b, 1 \leq l \leq k$ are skipped mismatches, except when $p_1 b = u(vu)b$ lies in $N^{[1]}$. We only count those that do not lie in the previous set, i.e. those that satisfy $l \geq k - l$. This leads to ϕ and $N^{(2)}$. Remark that $M^{(2)}$ is a multiset. One may have: $u(vu)w_2u(vu) = u(vu)[w_2'u(vu)]^2$ that rewrites $u'(v'u')^2$. Nevertheless, the associated quasi-mismatches are different; hence this word is to be counted twice.

We now proceed with the correction of $L^{(3)}$ and $L^{(4)}$. The expression for $M^{(3)}$ steadily derives from $L^{(3)}$. Finally, the correcting set $L^{(4)}$ is built on quasi-mismatches $a(wa)^l(wa)^m$ with $l, m \geq 1$ and $b \preceq wa$. Such quasi-mismatches are not skipped. Hence, $M^{(4)} = \emptyset$.

Lemma 7.2 *Define $S(z)$ and $P(z)$ as the generating functions of non-empty non-bordered words and primitive words. The generating function $M^{(1)}$ satisfies, in the uniform case:*

$$M_{\bar{b}}^{(1)}(z) = S(z) + \frac{z}{1-z} P(z) .$$

Additionnally,

$$S(z) = 1 + qz + q(q-1)\sum_{j=0}^{\infty}(-1)^j z^{2j+1}\prod_{i=0}^{j} W(z^{2^i}) \quad (5)$$

$$P(z) = \sum_{d}\mu(d)(W(z^d)-1) \quad (6)$$

where μ is the Möbius function.

Proof: Given vu in P, $u(vu)^{k-1}$ is an extension whose characters and length are known. I.e. each extension contributes by z^l. We get: $\sum_m p_m z^m(z + z^2 + \ldots) = \frac{z}{1-z}P(z)$. The closed formula for $P(z)$ is classical and the one for $S(z)$ is derived in [Rég92b]. Deriving a closed formula for other generating functions appear non trivial. Notably, the contribution to $M^{(2)}$ of words $w_1 w_1$ depends on properties of primitive words. As the results above prove the feasability of the approach, we leave this open and rather provide dominating terms for small m.

8 Bernoulli stationary distributions

We now summarize the results of the previous sections, when a general $(\{p_a\},\{q_a\})$ data distribution is assumed. Most of these results have been presented in [Rég89], for uniform distributions, and [Rég91] for biased distributions.

Morris-Pratt variant To get the exact results for a fixed m, we perform a Taylor expansion of our generating functions (see appendix). By truncation, we get the exact values for smaller m. We group our results for the uniform case in the table below. Values $2, 3, 6$ and 10 are associated to the first contributions of $L^{(1)}, L^{(2)}, L^{(3)}$ and $L^{(4)}$.

m	MP
2	$1 + q^{-1} - q^{-2}$
3	$1 + q^{-1} - q^{-3} - q^{-4} + q^{-5}$
6	$1 + q^{-1} - q^{-4} - 2q^{-6} + q^{-9} + q^{-10}$
10	$1 + q^{-1} - q^{-4} - q^{-6} - q^{-8} - q^{-10} + q^{-12} + q^{-13} + q^{-14} + q^{-15} - q^{-16} + q^{-18} - q^{-19}$
11	$1 + q^{-1} - q^{-4} - q^{-6} - q^{-8} - q^{-11} + q^{-13} + q^{-14} + 2q^{-15} - q^{-16} + q^{-17} + q^{-18} - 2q^{-19}$

Due to the rapidly growing size of the results in the biased case, that hardly fit in a table, we stop our example at $m = 6$. This result can be used to derive an approximation of C_m^{MP}, when $m \to \infty$. This means notably that m is large enough in order to drop σ_i^{m+1}. Also, the terms of the same order must be dropped. The approximation is valid if q is not too small. The validity domain clearly depends on the distribution: when some characters are very rare, the σ_i^k sequences may decrease very slowly. Finally:

Corollary 8.1 *For large m, we have:*

$$C_m^{MP} \sim C_6^{MP} + \sigma_1^6 - (\sigma_3^2 - \sigma_1\sigma_5 + \sigma_2\sigma_4)$$

and

$$\begin{aligned} C_6^{MP} &= 1 + \sigma_1 - \sigma_1\sigma_2 + \sigma_3 - {\sigma_1}^2\sigma_2 + \sigma_2^2 - \sigma_1^3\sigma_2 - \sigma_1\sigma_3 + \sigma_4 - \sigma_1\sigma_4 + \sigma_2\sigma_3 + \sigma_5 \\ &\quad -{\sigma_1}^6 - \sigma_1^4\sigma_2 + \sigma_3^2 - \sigma_1\sigma_2\sigma_3 + \sigma_2^3 - \sigma_1\sigma_5 + \sigma_2\sigma_4 \end{aligned} \quad (7)$$

For uniform distributions, this development can be expressed as a function of cardinality q:

$$1 + \frac{1}{q} - (\frac{1}{q^4} + \frac{1}{q^6} + \frac{1}{q^8}) .$$

Remark that here the results for smaller m derive by truncation of this final result. For instance, $C_3^{MP} = 1 + \sigma_1 - {\sigma_1}^3 - \sigma_1\sigma_2 + \sigma_3$.

Knuth-Morris-Pratt variant We provide costs up to $m = 6$. In the uniform case, the first contributing terms of $M^{(2)}, M^{(3)}$ and $N^{(2)}$ are:

$$\begin{aligned} M_6^{(2)}(z) &= (q-1)z[M^{(1)}(z^2)zW(z) + (qz^4 + S(z^2)) \\ &+q(z^3 + z^4 + 2z^5) + q(q-1)z^5] \quad (8) \\ M_6^{(3)}(z) &= q(q-1)z^6 \quad (9) \\ N_6^{(2)}(z) &= 0 \quad (10) \\ N_6^{(1)} &= q(q-1)^2z^6 \quad (11) \end{aligned}$$

We get the formula

```
C6KMP(z) = (-q^2+q)*z^2+(-q^3+2*q^2-q)*z^3+(-q^4+2*q^2+q^3-2*q)
         + (-q^2-q^5+2*q^3-q+q^4)*z^5+(-2*q^2+q^3-q^6+q^4+q^5)*
```

from which the constants for $m \leq$ derive by truncation and substitution. And, we get asymptotics on the constant for greater m. Also, in the biased case, $M^{(1)}_{\bar{b}} = z_{\bar{b}} + O(z^2_{\bar{b}})$ which yields the first contributing term: $\sum q_{\bar{b}} p_{\bar{b}} (1 - p_{\bar{b}}) \equiv -s_{2,1}$. Finally:

Corollary 8.2 *For large m, we have:*

$$C'^{KMP}_m \sim 1 + \sigma_1 - s_{2,1}$$

For uniform distributions, this development can be expressed as a function of cardinality q:

$$1 + \frac{1}{q} - \frac{1}{q^2} - \frac{1}{q^5} + \frac{1}{q^6} - \frac{1}{q^8} .$$

It is worth noticing that the development at the first order, $1 + \sigma_1$, is an upper bound for all variants, as the existence of a quasi-mismatch is a necessary condition for any extra-comparison. This proves notably that Morris-Pratt-like algorithms outperform the naive algorithm, despite the (false) paradox in [Bar85]. Remark that the result for Knuth-Morris-Pratt running under biased distributions, coincides, at the first order $1 + \sigma_1$, with the conjecture given in [BY89a] on the basis of an approximation of the Markov chain when $p_a = q_a$. Hence, it is supported by the simulations presented there. Since our first submission to this journal [Rég91], some results were found by [Han91]. Exact results are given for biased distributions and $m \leq 4$. For uniform distributions, the exact costs are given up to $m = 10$ that imply for greater m the asymptotic developments at order 4 given in our table above. We are grateful to the author for pointing out an error at order $m = 6$ in our asymptotic development.

9 Markov dependency

We group our results in Theorem 9.2, where an asymptotic development of the linearity constant is given. They only depend of the initial and stationary distributions, of the fundamental matrix of F and of length m. The order of approximation is a simple function of the distribution.

9.1 Extended formalism

Matrices formalism For k-Markovian distributions, multivariate generating functions are not so meaningful. All occurrences of a given character

are not equivalent, due to the correlation to the k previous characters. A matrix model appears more powerful. More precisely:

Definition 8 *Let $\mathcal{L}$ be a language on alphabet A, with probability measure $\mathcal{P}$ and $\mathcal{L}_{i,j} = \mathcal{L} \cap a_i.A^* \cap A^*.a_j$. Let $L^{(1)}_{\mathcal{P}}$ be the diagonal matrix:*

$$L^{(1)}_{\mathcal{P}} = ||p_i.1_{\mathcal{L}\cap a_i.A^*\neq 0}||_{i=1..q} \tag{14}$$

Let $L^{(2)}_{\mathcal{P}}$ be a matrix satisfying the equation:

$$||E_{\mathcal{P}}(\mathcal{L}_{i,j})|| = L^{(1)}_{\mathcal{P}} \circ L^{(2)}_{\mathcal{P}} \ . \tag{15}$$

Then $(L^{(1)}_{\mathcal{P}}, L^{(2)}_{\mathcal{P}})$ is a matricial expectation couple for $\mathcal{L}$.

Remark that when $\mathcal{P}$ changes to $\mathcal{Q}$, $L^{(i)}_{\mathcal{Q}}$ is derived from $L^{(i)}_{\mathcal{P}}$ by the set of substitutions: $p_{i,j} \rightarrow q_{i,j}, p_i \rightarrow q_i$. Similarly, one notes $L^{(i)}_{\mathcal{P}\mathcal{Q}}$ the matrix derived from $L^{(i)}_{\mathcal{Q}}$ by the substitutions: $q_{i,j} \rightarrow p_{i,j}q_{i,j}, q_i \rightarrow p_i q_i$.
Example: Assume a 1-Markov dependency characterized by $(||q_i||, Q = ||q_{i,j}||)$. Then, the expectation couple associated to A^{*+} is: $(||q_i||, (I-Q)^{-1})$.

To express our results, we also define a set of *operators*:

Definition 9 *Given $M = ||m_{i,j}||$ a matrix and a a character from the alphabet A, we define the operators:*

* $Line_a = ||m_{i,j}.1_{a_i=a}||$ and $Col_a = ||m_{i,j}.1_{a_j=a}||$.
* $\bar{Line}_a(M) = M - Line_a(M)$ and $\bar{Col}_a(M) = M - Col_a(M)$
* $S(M) = \sum_{i,j} m_{i,j}$

Finally, assume $m_{i,j} = \phi_{i,j}(x_1, \ldots, x_k)$. We note $M_m = ||\phi_{i,j}(x_1^m, \ldots, x_k^m)||$.

Remark: $Line_a(M \circ N) = Line_a(M) \circ N$ and $Col_a(N \circ M) = N \circ Col_a(M)$; $N \circ Line_a(M) = Col_a(N) \circ M$, and $N \circ \bar{Line}_a(M) = \bar{Col}_a(N) \circ M$. Practically, M_m will be used with variables x_k ranging on $(q_{i,j}), (p_{i,j}), (q_i)$ and (p_i).

Theorem 9.1 *Let $\mathcal{L}, \mathcal{M}$ and $\mathcal{N}$ be three languages on a q-alphabet A. We note $(L^{(1)}, L^{(2)}), (M^{(1)}, M^{(2)})$ and $(N^{[1]}, N^{(2)})$ their expectation couples when $\mathcal{P}$ is a k-order Markov dependency, and assume then the size of all words in $\mathcal{L}, \mathcal{M}, \mathcal{N}$ is greater than or equal to k. Then:*

(1) If $\mathcal{L} = \mathcal{M} \cup \mathcal{N}$ *and if* $\mathcal{L} \cap \mathcal{M} = \emptyset$: $(L^{(1)}, L^{(2)}) = (M^{(1)}.N^{[1]}, M^{(2)})$.

(2) If $\mathcal{L} = \mathcal{M} \cdot \mathcal{N}$ *and if the decomposition of* $\mathcal{L}$ *onto* $\mathcal{M} \cdot \mathcal{N}$ *is* unique:

$$(L^{(1)}, L^{(2)}) = (M^{(1)}, M^{(2)} \circ (P^{[2](k)} \cdot N^{[0]}) \circ N^{(2)})$$

(3) If $\mathcal{L} = \mathcal{M}^l, l \in N$: $(L^{(1)}, L^{(2)}) = (M^{(1)} \cdot P^{[2]}{}_{l-1}, M^{(2)}{}_l)$

9.2 Performance

Previous results do not easily generalize to Markovian distributions. A derivation similar to the derivation of $L^{(2)}$ is possible, using operators *Line* and *Col*. But the tedious computations are left to the interested reader. An example of such a computation can be found in [Rég92b]. We will rather prove a set of general upper bounds.

Lemma 9.1 *For k-order Markov dependencies:*

$$\sum_{|p|=m} E_{\mathcal{P}}(p) \sum_{w \in \mathcal{L}_p^{(1)}} E_{\mathcal{Q}}(w) = E_{\mathcal{QP}}(Diag(A \times A)) - E_{\mathcal{QP}}(Diag(A^m \times A^m)) \ ,$$

Proof: Now, we rewrite the expectation as: $E_{\mathcal{QP}}(Diag(A+\cdots+A^{m-1})^2).E_{\mathcal{Q}}(A) - E_{\mathcal{QP}}(Diag((A^2+\cdots+A^m)^2))$. The first term rewrites: $\sum_{j=1}^{m-1} E_{QP}(Diag(A^j \times A^j))$, and the second is the same sum where j is shifted by 1. By elimination again, we get: $E_{\mathcal{QP}}(Diag(A \times A) - E_{\mathcal{QP}}(Diag(A^m \times A^m)$.

Theorem 9.2 *Assume Markovian distributions for the text and the pattern. The average number of comparisons performed by Morris-Pratt algorithm is linear and the linearity constant satisfies:*

$$c \ = \ 1 + \sum_a q_a p_a - \mathcal{S}(S_{1,1} \circ Diag(F) \circ (I - Diag(F))^{-2} \circ F) + O(\alpha^5) \ .$$

where $\alpha = q.max(\sum p_{c,a} q_{c,a}, \sum p_c q_c)$.

Proof: We first derive the dominating term. From the definition, the expectation couple for A^j is $(S_{0,1}; Q^{j-1})$. Hence, the second term in $E_{\mathcal{QP}}(L^{(1)})$ is $O(\alpha^m)$. Here, 1-order dependencies increase the complexity of the computation as one must consider small words independently of the general case. Namely, to compute $\sum_{w \in \mathcal{L}_p^{(2)}} E_{\mathcal{Q}}(w)$, let $\mathcal{L} = \{c^{l+1}, c \in A, l \geq 1\}$. Repeatedly applying rule (2) yields the expectation couple $(Col_c(S_{0,1}); Col_c(Q)^l)$ for

a given c. Concatenation with $a \neq c$ yields a multiplication by $Q - Col_c(Q)$ or $Q - Diag(Q)$. Applying now our translation rules, we get:

$$\sum_{l \geq 1} l[S_{1,1} \circ Diag(F)^l \circ (F - Diag(F))] = S_{1,1} \circ Diag(F) \circ (I - Diag(F))^{-2} \circ (F - Diag(F)) .$$

We have neglected words of size greater than 1. The smallest neglected sequence is $b.cb.a, a \neq c$ whose expectation is upper bounded by:

$$\mathcal{S}(S_{1,1} \circ [Diag(F^2) - Diag(F)^2] \circ F) = O(\alpha^5) .$$

Notice that, to get an upper bound on the contribution of a neglected language, i.e. on its expectation, it is enough to forget some restrictive condition. Here,a rough approximation is to forget conditions on words u_i and $v_i 1$ and consider only the number of repetitions. Hence, our approximation order.

Further terms in the development can be obtained in a similar manner, e.g. by construction of the smaller words in sets $\mathcal{L}^{(i)}$. This will be interesting for very biased distributions. Notice that the choices and the expressions ofthese subsets have a great influence on the convergence rate of the asymptotic development of the linearity constant as well as the efficiency of its computation (notably it must not be exponential!). Finally, Knuth-Morris-Pratt performance are easily derived from above, by a slight modification of $\mathcal{L}_p^{(2)}$, as detailed for stationary distributions above.

10 Conclusion

We have presented an average analysis of Morris-Pratt-like string searching algorithms, assuming various probabilistic models: 1-order Markov dependency between characters, stationary models. We provide an answer to conjectures over the expected behavior. The expected number of comparisons was proved to be asymptotically kn, and linearity constants were derived. A closed formula was proved. An approach via word enumeration was proposed that proved to be powerful, as it "sticks" to the intrinseque nature of the algorithms. Also, it allows for using the powerful toolkit of generating functions. Boyer-Moore-like algorithms can be analysed in the same manner: it will be considered in a companion paper. A challenging problem is now to determine different range domains for m, q and the *data distributions* so as the best algorithm may be chosen in any case, and notably the pathological

distributions. The scheme should also apply to other algorithms that preprocess the pattern, such as the one in [CGG90] or multidimensional search [BYR93]. It is also worth extending that work to string prefix-matching [BCT93] or to string searching with k mismatches.

References

[Bar85] G. Barth. An analytical comparison of two string matching algorithms. *IPL*, 30:249–256, 1985.

[BCT93] D. Breslauer, L. Colussi, and L. Toniolo. Tight Comparison Bounds for the String Prefix-Matching Problem. In *CPM'93*, volume 684 of *Lecture Notes in Computer Science*, pages 11–19. Springer-Verlag, 1993. In Proc. 4-th Symposium on Combinatorial Pattern Matching, Padova, Italy.

[BM77] R. Boyer and S. Moore. A fast string searching algorithm. *CACM*, 20, 1977.

[Bre93] D. Breslauer. Testing superprimitivity. *IPL*, 44:345–347, 1993.

[BY89a] R. Baeza-Yates. Efficient text searching. PhD Thesis CS-89-17, Univ. Waterloo, Canada, 1989.

[BY89b] R.A. Baeza-Yates. String Searching Algorithms Revisited. In *WADS'89*, volume 382 of *Lecture Notes in Computer Science*, pages 75–96. Springer-Verlag, 1989. Proc. WADS'89, Ottawa.

[BYGR90] R. Baeza-Yates, G. Gonnet, and M. Régnier. Analysis of Boyer-Moore-type string searching algorithms. In *SODA'90*, pages 328–343. SIAM, 1990. Proc. Siam-ACM Symp. on Discrete Algorithms, San Francisco, USA.

[BYR92] R. Baeza-Yates and M. Régnier. Average running time of Boyer-Moore-Horspool algorithm. *Theoretical Computer Science*, 92:19–31, 1992. special issue.

[BYR93] R. Baeza-Yates and M. Régnier. Fast algorithms for two dimensional and multiple pattern matching. *IPL*, 45(1):51–57, 1993. Preliminary draft in Proc. Swedish Workshop on Algorithm Theory, Bergen, Norway, 1990.

[CGG90] L. Colussi, Z. Galil, and R. Giancarlo. On the exact Complexity of string matching. In *FOCS'90*, pages 135–143. IEEE, 1990. Proc. 31-st Annual IEEE Symposium on the Foundations of Computer Science.

[Eil74] Samuel Eilenberg. *Automata, Languages, and Machines*, Volume A. Academic Press, 1974.

[Han89] Ch. Hancart. Sur le cas moyen des algorithmes de recherche d'un mot dans un texte. DEA, Université de Paris VII, 1989.

[Han91] Ch. Hancart. Algorithme de Morris et Pratt et ses raffinements: une analyse en moyenne. Research report 91.56, Université de Paris VII, October, 1991.

[Han93] Ch. Hancart. *Analyse Exacte et en Moyenne d'Algorithmes de Recherche d'un Motif dans un Texte.* Univ. Paris-VI, Paris, France, 1993. Thèse de 3eme cycle, to appear.

[Hor80] R. N. Horspool. Practical fast searching in strings. *Software-Practice and Experience*, 10:501–506, 1980.

[HU79] J. E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory.* Addison Wesley, Reading, Mass, 1979.

[KMP77] D.E. Knuth, J. Morris, and V. Pratt. Fast pattern matching in strings. *SIAM J. on Computing*, 6:323–350, 1977.

[KR87] R. Karp and M. Rabin. Efficient randomized pattern-matching algorithms. *IBM J. Res. Development*, 31:249–260, 1987.

[Lot83] Lothaire. *Combinatorics on Words.* Addison-Wesley, Reading, Mass., 1983.

[Rég89] M. Régnier. Knuth-Morris-Pratt algorithm: an analysis. In *MFCS'89*, volume 379 of *Lecture Notes in Computer Science*, pages 431–444. Springer-Verlag, 1989. Proc. Mathematical Foundations for Computer Science 89, Porubka, Poland.

[Rég91] M. Régnier. Performance of String Searching Algorithms under Various Probabilistic Models, 1991. INRIA Research Report 1565.

[Rég92a] M. Régnier. Enumeration of bordered words. *RAIRO Theoretical Informatics and Applications*, 26,4:303–317, 1992.

[Rég92b] M. Régnier. A language approach to string searching evaluation. In *CPM'92*, volume 644 of *Lecture Notes in Computer Science*, pages 15–26. Springer-Verlag, 1992. Proc. 3-rd Symposium on Combinatorial Pattern Matching, Tucson, Arizona.

[Riv77] R. L. Rivest. On the Worst-Case Behavior of String-Searching Algorithms. *S.I.A.M. J. on Comp.*, 6:669–674, 1977.

[RS94] M. Régnier and W. Szpankowski. Exact Complexity of Sequential Pattern Matching Algorithms, 1994. in preparation.

[Sch88] R. Schaback. On the Expected Sublinearity of the Boyer-Moore Algorithm. *SIAM J. on Computing*, 17:548–558, 1988.

[Tho68] K. Thompson. Regular expression search algorithm. *CACM*, 11:419–422, 1968.

[VF90] Jeffrey Scott Vitter and Philippe Flajolet. Analysis of algorithms and data structures. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume A: Algorithms and Complexity, chapter 9, pages 431–524. North Holland, 1990.

[Yao79] A. C. Yao. The complexity of pattern matching for a random string. *SIAM J on Computing* ., 8:368–387, 1979.

Unité de Recherche INRIA Rocquencourt
Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 LE CHESNAY Cedex (France)

Unité de Recherche INRIA Lorraine Technopôle de Nancy-Brabois - Campus Scientifique
615, rue du Jardin Botanique - B.P. 101 - 54602 VILLERS LES NANCY Cedex (France)
Unité de Recherche INRIA Rennes IRISA, Campus Universitaire de Beaulieu 35042 RENNES Cedex (France)
Unité de Recherche INRIA Rhône-Alpes 46, avenue Félix Viallet - 38031 GRENOBLE Cedex (France)
Unité de Recherche INRIA Sophia Antipolis 2004, route des Lucioles - B.P. 93 - 06902 SOPHIA ANTIPOLIS Cedex (France)

EDITEUR
INRIA - Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 LE CHESNAY Cedex (France)

ISSN 0249 - 6399